RSimplex: A Robust Control Architecture for Cyber And Physical Failures

XIAOFENG WANG, University of South Carolina
NAIRA HOVAKIMYAN and LUI SHA, University of Illinois at Urbana-Champaign

As the complexity of Cyber-Physical Systems (CPS) increases, it becomes increasingly challenging to ensure CPS reliability, especially in the presence of software and/or physical failures. The Simplex architecture is shown to be an efficient tool to address software failures in such systems. When physical failures exist, however, Simplex may not function correctly because physical failures could change system dynamics and the original Simplex design may not work for the new faulty system. To address concurrent software and physical failures, this article presents the RSimplex architecture, which integrates Robust Fault-Tolerant Control (RFTC) techniques into the Simplex architecture. It includes the uncertainty monitor, the High-Performance Controller (HPC), the Robust High-Assurance Controller (RHAC), and the decision logic that triggers the switch of the controllers. Based on the output of the uncertainty monitor, we introduce a monitor-based switching rule in the decision logic in addition to the traditional envelope-based rule. The RHAC is designed based on RFTCs. We show that RSimplex can efficiently handle a class of software and physical failures.

Categories and Subject Descriptors: D.4.5 [**Software Engineering**]: Reliability—*Fault-tolerance*; I.2.8 [**Computing Methodologies**]: Problem Solving, Control Methods, and Search—*Control theory*

General Terms: Design, Reliability

Additional Key Words and Phrases: Simplex architecture, fault-tolerant control, cyber and physical failures

ACM Reference format:
Xiaofeng Wang, Naira Hovakimyan, and Lui Sha. 2018. RSimplex: A Robust Control Architecture for Cyber And Physical Failures. *ACM Trans. Cyber-Phys. Syst.* 2, 4, Article 27 (July 2018), 26 pages.
https://doi.org/10.1145/3121428

1 INTRODUCTION

Cyber-Physical Systems (CPS) consist of two main components: *physical elements* modeling the systems to be controlled and *cyber elements* representing the communication links and the comp utation software. Such systems are ubiquitous in a number of application areas, including aircraft and air-traffic control, highway transportation, space, healthcare, medicine, and manufacturing, to name a few. As the control missions become increasingly complex in real life, CPS architectures

This work is supported by the National Science Foundation under grant CNS-1330077 and CNS-1329870.
Authors' addresses: X. Wang is with the Department of Electrical Engineering at the University of South Carolina, Columbia, SC, 29208; email: wangxi@cec.sc.edu; N. Hovakimyan is with the Department of Mechanical Science and Engineering at the University of Illinois at Urbana-Champaign, Urbana, IL 61801; email: nhovakim@illinois.edu; L. Sha is with the Department of Computer Science at the University of Illinois at Urbana-Champaign, Urbana, IL 61801; email: lrs@illinois.edu.

https://doi.org/10.1145/3121428

in turn become increasingly complicated, which makes it very challenging to guarantee the safety of CPS, especially in the presence of both cyber and physical failures. So there is a great need for tools that can handle failures in CPS.

When physical damage is coupled with cyber failures, it raises a unique challenge with respect to system safety. Historically, Robust Fault-Tolerant Control (RFTC) theory is developed largely independent of software assurance technologies, and vice versa. Although significant progress has been made to address either software or physical failures, few of these approaches can work in conjunction to counter "concurrent" software and physical failures because they are developed under different assumptions and models. By "concurrent," we mean that software failures and physical failures may happen at the same time. On the one hand, software assurance technologies are model-based and require a profile of the physical dynamics. When the physical dynamics change due to damage and failures, the pre-stored profiles will become invalid and the original assurance approach may not be able to function correctly. On the other hand, though existing RFTC techniques can efficiently compensate for physical damage, it is critical to guarantee that the control software is not compromised. Given a complex system, it is almost implausible, if not impossible, to overcome all potential software failures under different physical models led by physical failures. It is a prime time to develop unified models with coherent set of assumptions, supported by integrated technologies, upon which concurrent cyber and physical failures can be addressed in a much more effective way.

With this observation, this article tackles two issues: (*i*) how to detect (at least a class of) software failures and physical failures and (*ii*) how to isolate or minimize the negative impact of failures on the control system once detected. Our solution is the RSimplex (Robust Simplex) architecture that includes a High-Performance Controller (HPC), a Robust High-Assurance Controller (RHAC), an uncertainty monitor, and decision logic. It is assumed that, in RSimplex, software failures only happen in the HPC, which can be any complex controller providing high levels of performance and advanced functionalities, but which may not be (fully) verified and can experience software faults. The other components in RSimplex must be fully verified. Under nominal operating conditions, the HPC is activated. The uncertainty monitor and the decision logic are used to detect physical and software failures, respectively. Once detected, control is switched from the HPC to the RHAC.

To detect physical failures, the uncertainty monitor estimates the scale of uncertainty in the system due to those failures. We show that the output of this monitor is equal to low-pass filtered uncertainty. Based on the monitor, a monitor-based switching rule is proposed in addition to the traditional envelope-based switching rule; this triggers the switch from the HPC to the RHAC. To be more specific, the decision logic states that the controller must switch to the RHAC when (*i*) the estimated scale of physical failures computed by the uncertainty monitor exceeds a prespecified threshold, or (*ii*) the state is on the boundary of the safety envelope and about to leave this envelope.

A critical requisite in RSimplex is that, at the switching moment, the state must stay in the invariant set of the RHAC. Then the RHAC can always keep the state inside this invariant set after switching. If this invariant set is also inside the state constraint set, the RHAC will be able to guarantee the satisfaction of the state constraints. Based on this idea, we use the invariant set of the RHAC as the safety envelope in the switching rule. To obtain a computable invariant set, the RHAC must provide predictable transient performance even in the presence of physical failures. We propose two different types of RFTCs to fulfill this requirement: monitor-based RHAC and $\mathcal{L}_1$-based RHAC. The former is designed based on the output of the uncertainty monitor, and the latter is basically an $\mathcal{L}_1$ adaptive controller (Hovakimyan and Cao 2010). We prove that, with either RHAC, the system state will follow the state of an ideal model (which is completely known) within a small and uniformly bounded bias, even in the presence of consecutive physical failures. This property enables us to compute the safety envelope based on the ideal model and the bound on the bias. Furthermore, we present an approach to extend the size of the safety envelope using multiple RHAC candidates. A preliminary version was presented in Wang et al. (2013) called L1Simplex. Compared with the work in Wang et al. (2013), the physical failures are extended to be nonlinear. In addition, the monitor-based RHAC and a novel design of the uncertainty monitor are introduced that lead to new performance analysis. We also provide a sufficient condition to ensure the non-emptiness of the safety envelope.

This article is organized as follows. Section 2 discusses prior approaches addressing software and physical failures. Section 3 formulates the problem. The RSimplex architecture is presented in Section 4. An illustrative example is presented in Section 5. Section 6 draws conclusions and proposes future work. The proofs are in Section 7.

2 RELATED WORK

Among solutions for software reliability, two basic approaches play important roles. One is to avoid faults during the development of software by using formal methods and model-checking tools (Clarke et al. 1999; Baier et al. 2008). However, if the size of the system is large, it is impossible to have formal verification of every component. The other approach is to design real-time fault-tolerant software through diversity (Chen and Avizienis 1978). The Simplex architecture is one such fault-tolerant software tool, "using simplicity to control complexity" (Sha 2001). It was designed to tolerate specification faults, design faults, and implementation (logical) faults in the software of any complex HPC that provides high levels of performance and advanced functionalities. The basic idea of Simplex is described as follows: When a software failure occurs, Simplex switches the controller from the HPC to a High-Assurance Controller (HAC) that is fully certified and ensures safe and stable operation of the system with lower performance (Avizienis 1995; Sha 1998). This method has been extended to the NetSimplex architecture for networked control systems (Yao et al. 2013) and the ORTEGA scheme that runs HAC and HPC as separate tasks in one processor (Liu et al. 2008). Further extensions include the system-level Simplex architecture (Bak et al. 2009), the S3A architecture for enhanced security and robustness of CPS (Mohan et al. 2013), and real-time reachability analysis for Simplex design (Bak et al. 2014), to name a few.

Simplex turns out to be a very efficient tool to deal with software failures (Crenshaw et al. 2007; Seto and Marz 2000). However, when physical failures happen, Simplex architecture may not be able to function correctly. This is because Simplex is designed based on a given model of the physical dynamics. When the dynamics change due to physical failures, the original Simplex design may not be suitable for the new dynamics if the safety envelope for the old model is no longer valid. It is not clear how to design a controller switching rule and HAC that are capable of accommodating a large family of uncertain physical models.

On the other hand, to deal with physical failures, researchers seek RFTC techniques that provide high levels of performance and robustness in the presence of physical failures, such as Input-to-State Stable (ISS) control (Sontag and Wang 1995), $\mathcal{L}_1$ adaptive control (Hovakimyan and Cao 2010; Hovakimyan et al. 2011), model reference adaptive control (Åström and Wittenmark 2013), disturbance-observer control (Shim and Jo 2009), and internal model-based control (Harnefors and Nee 1998), to name a few. These approaches can either limit or minimize the effect of the physical failures. To ensure that the RFTC functions properly, however, the control software cannot fail. Otherwise, wrong control inputs will be actuated and the system will become unstable. In summary, prior work only focuses on addressing one type of failures, either cyber or physical. To guarantee safety of a complex CPS, we need novel approaches that can overcome concurrent cyber and physical failures.

3 PROBLEM FORMULATION

3.1 Notations

We denote by $\mathbb{R}^n$ the n-dimensional real vector space, and by $\mathbb{R}^+$ the set of the real positive numbers. Let $\mathbb{R}_0^+ = \mathbb{R}^+ \cup \{0\}$. The Euclidean norm of a vector x is denoted by $\|x\|$ and the induced 2-norm of a matrix A is denoted by $\|A\|$. The identity matrix is denoted as $\mathbb{I}$. Given a matrix P, $\lambda_{\max}(P)$ and $\lambda_{\min}(P)$ are the maximum and minimum singular value of P, respectively. The $\mathcal{L}_1$ norm of a function $x(t)$ is denoted by $\|x\|_{\mathcal{L}_1}$. The truncated $\mathcal{L}_\infty$ norm of $x(t)$ is defined as $\|x\|_{\mathcal{L}_\infty^{[a,b]}} = \sup_{a \le t \le b} \|x(t)\|$. The Laplace transform of a function $x(t)$ is denoted as $x(s) = \mathfrak{L}[x(t)]$, where we simply replace the time argument "t" of the function x by "s". Given a function $f : \mathbb{R}^n \to \mathbb{R}$, $\nabla f(x) \in \mathbb{R}^{1\times n}$ is the gradient of $f(x)$ at x.

3.2 System Model

This article studies fault-tolerant CPS, where the physical dynamics may change due to failures. We assume that the physical plant under the normal condition is governed by the following state equation:

$$\begin{aligned} \mathcal{P}_0 :\ & \dot{x}(t) = Ax(t) + Bu(t) + f_0(x,t), \\ & x(t_0) = x_0, \end{aligned} \tag{1}$$

where $x : \mathbb{R}_0^+ \to \mathbb{R}^n$ is the system state, $u : \mathbb{R}_0^+ \to \mathbb{R}^m$ is the control input, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ are known matrices, $x_0 \in \mathbb{R}^n$ is the initial state, and $f_0 : \mathbb{R}^n \times \mathbb{R}_0^+ \to \mathbb{R}^n$ is an unknown nonlinear function describing uncertainty in the system dynamics. For the nominal plant, $f_0(x(t),t)$ should be small.

When physical failures happen, the nominal dynamics $\mathcal{P}_0$ will change. Without loss of generality, we assume that there are totally N physical failures, and the kth physical failure occurs at time t_k, which results in the kth change in the physical dynamics. As a result, it generates a sequence of faulty physical dynamics $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_N$, where the dynamics of $\mathcal{P}_k$ are described as follows:

$$\mathcal{P}_k :\ \dot{x}(t) = Ax(t) + Bu(t) + f_k(x(t),t). \tag{2}$$

The initial condition of $\mathcal{P}_k$ is $x(t_k)$. The function $f_k : \mathbb{R}^n \times \mathbb{R}_0^+ \to \mathbb{R}^n$ is unknown. Usually physical systems have constraints on the system state, which can be described as $a_i^\top x \le 1$ for $i = 1, 2, \ldots, q$ with a known vector $a_i \in \mathbb{R}^n$. This constraint is called an *admissible constraint.* The combination of these inequalities admits a constraint set

$$\Omega = \{x \in \mathbb{R}^n \mid a_i^\top x \le 1, i = 1, 2, \ldots, q\}. \tag{3}$$

The system is *safe* if the system state always stays inside Ω. The *safety envelope* is a subset of Ω to be determined later.

The control software failures considered in this article are generic, including failures that prevent the system from obtaining the appropriate control inputs on time (i.e., out of memory, large computational delays, and generating inappropriate control signals). When the regular controller is experiencing these software failures, it may not be able to function correctly. An intuitive solution is to have a safety controller as an alternative, which will be activated when software failures happen, as shown in Figure 1. The resulting issues are how to detect potential software/physical failures and how to design the safety controller. Overall, the problem we attempt to tackle is how to co-design the control algorithm and the control software architecture such that system safety can

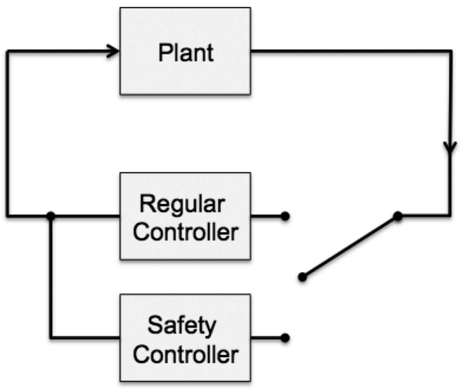

Fig. 1. A control architecture with a safety controller.

be guaranteed (i.e., $x(t) \in \Omega$ for any $t \geq 0$, even in the presence of physical failures and software failures).

3.3 Assumptions

There are several assumptions on the system dynamics.

Assumption 3.1 (Bounded Uncertainty). *Assume that $f_k(x,t)$ is uniformly bounded in t and Lipschitz in x over the set $x \in \Omega$ for any $k = 0, 1, \ldots, N$; that is, there exist positive constants l_k, b_k such that*

$$\|f_k(x_1,t) - f_k(x_2,t)\| \leq l_k \|x_1 - x_2\|, \tag{4}$$

$$\|f_k(0,t)\| \leq b_k \tag{5}$$

hold for any $t \geq 0$, $x_1, x_2 \in \Omega$, and $k = 0, 1, \ldots, N$. Also assume that l_k and b_k are known.

Remark 3.1. Notice that l_0 and b_0 will be much smaller than l_k and b_k, respectively, for $k = 1, 2, \ldots, N$ because the uncertainty in the nominal plant should be much smaller than in faulty plants.

Assumption 3.2 (Bounded Failure Rate). *There exists a lower bound on the inter-failure time intervals; that is, there exists a positive constant T such that $t_{k+1} - t_k \geq T$ holds for $k = 0, 1, 2, \ldots, N$.*

Remark 3.2. This assumption implies that the dynamics cannot switch too fast. It is necessary to ensure the existence of the solution to Equations (1) and (2). Otherwise, Zeno behavior may occur, which is beyond the scope of this article.

Assumption 3.3. *There are no sensor and actuator failures.*

Remark 3.3. In this article, we only focus on the class of physical failures whose effect can be modeled by f_k such as $f_k(x,t) = x^2 \sin t$.

3.4 An Illustrative Example

To better illustrate the problem, we consider an Inverted Pendulum (IP) as an example. The linearized dynamics of the IP is described as follows:

$$\begin{pmatrix} \dot{x}_p \\ \ddot{x}_p \\ \dot{\theta}_p \\ \ddot{\theta}_p \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I_p+m_p l_p^2)b_p}{d_p} & \frac{m_p^2 g l_p^2}{d_p} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-m_p l_p b_p}{d_p} & \frac{m_p g l_p (M_p+m_p)}{d_p} & 0 \end{pmatrix} \begin{pmatrix} x_p \\ \dot{x}_p \\ \theta_p \\ \dot{\theta}_p \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{I_p+m_p l_p^2}{d_p} \\ 0 \\ \frac{m_p l_p}{d_p} \end{pmatrix} u_p + f_0(x_p, \dot{x}_p, \theta_p, \dot{\theta}_p) \tag{6}$$

where x_p is the cart's position, θ_p is the pendulum's angle with respect to the vertical, g is gravitational acceleration, M_p is the mass of the cart, m_p is the mass of the pendulum, b_p is the friction of the cart, I_p is the inertia of the pendulum, l_p is the length to the pendulum's center of mass, u_p is the force applied to cart, and $d_p = I_p(M_p + m_p) + M_p m_p l_p^2$. The function f_0 is the modeling error from linearization.

This linearized model is obtained based on the assumption that $\sin(\theta)_p \approx \theta_p$, $\cos(\theta_p) \approx -1$, and $\dot{\theta}_p^2 \approx 0$ (in this case, the modeling error f_0 is very small). To ensure this property, for example, θ_p can be restricted over the interval from -0.3 radians to 0.3 radians and $-0.1 \leq \dot{\theta}_p \leq 0.1$. Also, the position x_p usually will be restricted over a finite region. These constraints define the constraint set Ω where the state of the IP must stay.

Assume that the IP is controlled by an HPC where the control performance can be measured by a cost function related to the states and control inputs. A software failure can be any failures that delay and/or modify the appropriate control inputs. A physical failure, for example, can be some damage on the wheels of the cart. Such damage will increase the friction coefficient b_p. Let the new friction coefficient be $\hat{b}_p$. Then the faulty model can be written as

$$\begin{pmatrix} \dot{x}_p \\ \ddot{x}_p \\ \dot{\theta}_p \\ \ddot{\theta}_p \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I_p+m_p l_p^2)b_p}{d_p} & \frac{m_p^2 g l_p^2}{d_p} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-m_p l_p b_p}{d_p} & \frac{m_p g l_p (M_p+m_p)}{d_p} & 0 \end{pmatrix} \begin{pmatrix} x_p \\ \dot{x}_p \\ \theta_p \\ \dot{\theta}_p \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{I_p+m_p l_p^2}{d_p} \\ 0 \\ \frac{m_p l_p}{d_p} \end{pmatrix} u_p$$
$$+ \underbrace{f_0(x_p, \dot{x}_p, \theta_p, \dot{\theta}_p) + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{-(I_p+m_p l_p^2)(\hat{b}_p-b_p)}{d_p} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{-m_p l_p(\hat{b}_p-b_p)}{d_p} & 0 & 0 \end{pmatrix} \begin{pmatrix} x_p \\ \dot{x}_p \\ \theta_p \\ \dot{\theta}_p \end{pmatrix}}_{f_1(x_p, \dot{x}_p, \theta_p, \dot{\theta}_p)}. \tag{7}$$

4 RSIMPLEX ARCHITECTURE

This section introduces the RSimplex architecture for CPS. As shown in Figure 2, the RSimplex architecture includes the HPC, the RHAC, the uncertainty monitor, and the decision logic:

— *High-Performance Controller (HPC)*: The HPC (the regular controller in Figure 1) can be any complex controller providing high levels of performance and advanced functionalities, which is active during normal operation of the system. However, it may not be (fully) verified and may experience software faults.

— *Robust High-Assurance Controller (RHAC)*: The RHAC is a *simple* and *verified* robust controller that ensures safe and stable operation of the system but provides limited levels of performance and reduced functionalities.

— *Uncertainty Monitor*: This *verified* monitor provides estimates of the uncertainties inside the system with fast adaptation, which takes the form of the state predictor in the $\mathcal{L}_1$ adaptive control architecture.

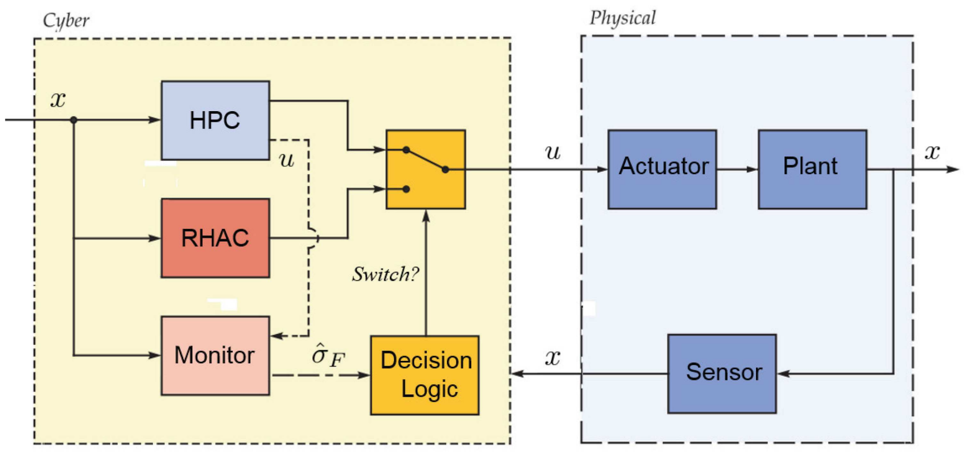

Fig. 2. The RSimplex architecture.

— *Decision Logic*: This logic, which must be *verified*, is responsible for switching from the complex HPC to the verified RHAC in the event of failures.

In this CPS framework, the sensors collect data on state information and transmit them to the controller. Under nominal operating conditions, the HPC is used, which provides desired performance with reduced operator workload. Meanwhile, the uncertainty monitor runs in an open-loop manner to estimate the present uncertainty. The estimate will be used as the triggering signal for a controller switch. The associated rule is that when the estimate exceeds a prespecified time-varying threshold, the controller must be switched from the HPC to the RHAC. The other switching rule is the traditional envelope-based rule: When the state is about to leave the safety envelope, execution of the HPC should be terminated and the RHAC will automatically be activated. No matter which controller is activated, the control inputs generated by the activated controller will be transmitted to the actuators for actuation.

We assume that, in RSimplex, software failures only happen in HPC. The RHAC, uncertainty monitor, and decision logic are fully verified and free of software failures. The following subsections discuss the design of the components in RSimplex.

4.1 The Uncertainty Monitor

Let (A_z, B_z, C_z) be the state space realization of the $n \times n$ matrix of stable and proper low-pass filters. The monitor is designed as follows:

$$\dot{z}(t) = A_z z(t) + (A_z B_z - B_z A)x(t) - B_z B u(t) \tag{8a}$$

$$\hat{\sigma}_F(t) = C_z z(t) + C_z B_z x(t) \tag{8b}$$

$$z(t_0) = -B_z x(t_0), \tag{8c}$$

where $z : \mathbb{R}_0^+ \to \mathbb{R}^l$ and $\hat{\sigma}_F : \mathbb{R}_0^+ \to \mathbb{R}^m$ are the state and the output of the monitor, respectively. The signal $\hat{\sigma}_F(t)$ will be used as a measurement of the uncertainty, which, as presented in Theorem 4.1, is equal to low-pass filtered uncertainty. Before showing such equivalence, we first define the uncertain signal $\sigma(t)$ inside the system:

$$\sigma(t) = \begin{cases} f_0(x(t), t), & t \in [t_0, t_1) \\ \dots \\ f_{N-1}(x(t), t), & t \in [t_{N-1}, t_N) \\ f_N(x(t), t), & t \in [t_N, +\infty) \end{cases}. \tag{9}$$

With the low-pass filter defined by (A_z, B_z, C_z), the low-pass filtered uncertainty $\sigma_F(t)$ is

$$\dot{z}_\sigma(t) = A_z z_\sigma(t) + B_z \sigma(t) \tag{10a}$$

$$\sigma_F(t) = C_z z_\sigma(t) \tag{10b}$$

$$z_\sigma(t_0) = 0 \tag{10c}$$

Theorem 4.1. *Given the system in Equation (1), the output of the monitor in Equation (8) is equal to the low-pass filtered uncertainty $\sigma_F(t)$ defined by Equation (10); that is, $\hat{\sigma}_F(t) = \sigma_F(t)$ for any $t \geq t_0$.*

Remark 4.2. Notice that the equivalence between $\hat{\sigma}_F(t)$ and $\sigma_F(t)$ is independent of the inputs generated by the HPC. So, even if the HPC generates the wrong control inputs due to software failures, the uncertainty monitor can still output correct estimates of the filtered physical uncertainty.

Theorem 4.1 implies that we can monitor the low-pass filtered uncertainty $\sigma_F(t)$ through the signal $\hat{\sigma}_F(t)$. Once a physical failure happens that leads to a change of physical dynamics, $\sigma(t)$ will become large. Then $\sigma_F(t)$ and $\hat{\sigma}_F(t)$ become large. Based on this observation, although we cannot directly measure the uncertainty $\sigma(t)$, we are able to detect severe physical failures by monitoring $\hat{\sigma}_F(t)$. If $\hat{\sigma}_F(t)$ exceeds a certain prespecified threshold, we understand that physical failures have occurred, and the controller will be switched to the RHAC. The choice of the threshold will be discussed in Section 4.2.

4.2 The Decision Logic

The controller switch can be triggered by different events. One is to check whether the output of the HPC is generated on time and is of the correct data type. If not, a switch takes place. Even if this requirement is met, it does not imply that the system is free of failures. We still need to monitor the internal signals for a possible switch. RSimplex provides two logic rules, either of which may lead to the transition from the HPC to the RHAC.

RULE I: The first rule is to switch when the signal $\hat{\sigma}_F(t)$ exceeds a threshold, as mentioned in Section (4.1). By the definition of $\sigma(t)$ in Equation (9), we know

$$\|\sigma(t)\| = \|f_0(x(t), t)\| \leq l_0\|x(t)\| + b_0, \quad \forall t \in [t_0, t_1)$$
$$\|\sigma(t)\| = \|f_k(x(t), t)\| \leq l_k\|x(t)\| + b_k, \quad \forall t \in [t_k, t_{k+1}), k = 1, 2 \dots, N.$$

Based on Equation (10), we have

$$\sigma_F(t) = C_z \int_0^t e^{A_z(t-\tau)} B_z \sigma(\tau) d\tau. \tag{11}$$

Therefore, for any $t \in [t_0, t_1)$,

$$\|\sigma_F(t)\| \leq \int_0^t \|C_z e^{A_z(t-\tau)} B_z\| \left(l_0\|x(\tau)\| + b_0\right) d\tau.$$

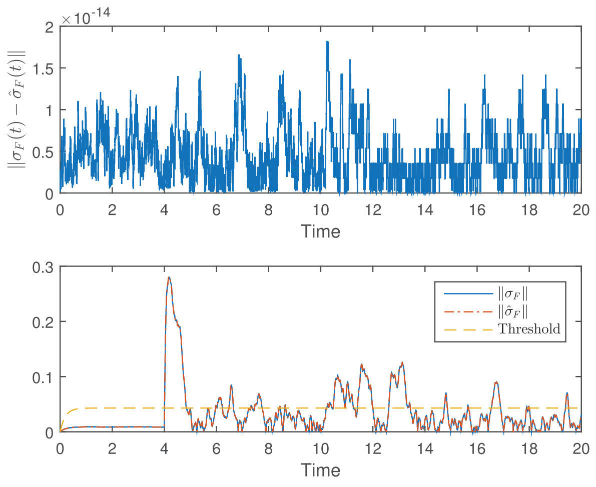

Fig. 3. Signals in the uncertainty monitor.

Since $\sigma_F(t) = \hat{\sigma}_F(t)$ by Theorem 4.1, we have that

$$\|\hat{\sigma}_F(t)\| \leq \int_0^t \|C_z e^{A_z(t-\tau)} B_z\| (l_0\|x(\tau)\| + b_0) d\tau \tag{12}$$

holds as long as $t \in [0, t_1)$. Once $\hat{\sigma}_F(t)$ exceeds the state-dependent threshold at the right-hand side of the preceding inequality, it implies that $t \notin [0, t_1)$ and therefore failures occur. As a result, we can use the violation of inequality (Equation (12)) to trigger the controller switch. Obviously, this rule is designed to detect physical failures since it is based on the size of the uncertainty.

Example 4.3. Consider the example of IP in Section 3.4. We run the uncertainty monitor where the control inputs of the HPC are randomly generated over $[0, 6]$. The friction coefficient is changed to 10 at $t = 4$. Figure 3 shows the history of $\|\sigma_F(t) - \hat{\sigma}_F(t)\|$ (top) and $\|\hat{\sigma}_F(t)\|$ (bottom). It can be found from the top plot that $\|\sigma_F(t) - \hat{\sigma}_F(t)\|$ is almost zero even under random control inputs (it is nonzero in the plot because of the numerical error in the simulation). In the bottom plot, $\|\hat{\sigma}_F(t)\|$ exceeds the threshold almost instantly after the physical failure occurs at $t = 4$. The accuracy of the monitor is independent of the control input. Simulations in Section 5 also verify this observation.

RULE II: The other rule adopts the same idea as the prior work on Simplex, where the controller switches when the state is leaving the prespecified safety envelope. However, the choice of the envelope for RSimplex is different. Let us start the design from the invariant set of an ideal model defined by

$$\dot{x}_{\text{id}}(t) = Ax_{\text{id}}(t) + Bu_{\text{id}}(t) \tag{13a}$$

$$u_{\text{id}}(t) = Kx_{\text{id}}(t), \tag{13b}$$

where $x_{\text{id}} : \mathbb{R}_0^+ \to \mathbb{R}^n$ and $u_{\text{id}} : \mathbb{R}_0^+ \to \mathbb{R}^m$ are the ideal state and input, respectively, and $K \in \mathbb{R}^{m\times n}$ is the nominal feedback ensuring that $A_m = A + BK$ is Hurwitz. Therefore, there must exist two positive-definite matrices $P, Q \in \mathbb{R}^{n\times n}$ such that

$$PA_m + (A_m)^\top P = -Q. \tag{14}$$

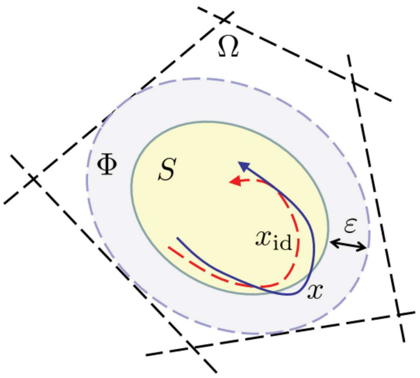

Fig. 4. The safety envelope of RSimplex.

Note that, compared with the plant in Equation (1), this model does not contain the uncertainty $f_0(x(t), t)$. Thus, it is called an "ideal" model. Given this model, we can identify its invariant set as

$$\Phi = \{x \in \mathbb{R}^n \mid x^\top P x \leq 1\}. \tag{15}$$

Once the ideal state $x_{\rm id}(t)$ is in this set Φ, it will never leave Φ. The next lemma ensures that this invariant set is inside the constraint set Ω.

LEMMA 4.4 (SETO AND SHA 1999). *Given the ideal model in Equation (13), $\Phi \subseteq \Omega$ if and only if $a_i^\top P^{-1} a_i \leq 1$, $i = 1, 2, \ldots, q$.*

This lemma implies that the matrix P must satisfy not only Equation (14), but $a_i^\top P^{-1} a_i \leq 1$ as well. These constraints on P can be presented as Linear Matrix Inequalities (LMIs). How to maximize the volume of Φ, subject to these LMIs can be found in Seto and Sha (1999).

With the set Φ in Equation (15), we can define the safety envelope S for RSimplex. Given a positive constant ε, the safety envelope is an ellipsoid obtained by shrinking Φ by ε:

$$S = \left\{x \in \mathbb{R}^n \mid x^\top P x \leq \gamma \text{ and } \min_{y \in \partial\Phi} \|x - y\| \geq \varepsilon \right\}, \tag{16}$$

where the set $\partial\Phi$ is the boundary set of Φ defined by $\partial\Phi = \{y \in \mathbb{R}^n \mid y^\top P y = 1\}$ and $\gamma < 1$. The basic idea is shown in Figure 4.

Obviously, $S \subset \Phi$ is also an invariant set of the ideal model. Intuitively, if we can guarantee that, at the switching moment T^* (which is not necessarily t_1) the state $x(T^*)$ is inside S, and

$$\|x(t) - x_{\rm id}(t)\| \leq \varepsilon \tag{17}$$

holds after switch, then $x(t)$ will always be inside Φ, since $x_{\rm id}(t) \in S$ for any $t \geq T^*$ with the initial state $x_{\rm id}(T^*) = x(T^*)$. How to select ε and enforce inequality (Equation (17)) will be discussed in Section 4.3.

Another observation is that the constant ε cannot be arbitrarily large. Otherwise, the set S could be empty. The existence of a non-empty S is established as follows:

LEMMA 4.5. *The set S is non-empty, if ε satisfies*

$$\varepsilon < \frac{1}{\lambda_{\max}(P)}, \tag{18}$$

where $\lambda_{\max}(P)$ is the maximum singular value of P.

Proof. The proof is straightforward and is therefore omitted. □

With the set S, we can mathematically define the envelope-based switching rule: The controller switches when

$$x(t)^\top P x(t) = \gamma \ \text{ and } \ 2x(t)^\top P \dot{x}(t) > 0. \tag{19}$$

The first equation implies that the state is on the boundary of S, and the second inequality indicates that the direction of the state's movement is pointing out of S.

4.3 RHAC Design

This section introduces the design of the RHAC. As mentioned in the previous subsection, in order to make Rule II valid, the RHAC must guarantee the satisfaction of inequality (Equation (17)) after the switch. The following discussion will present two different types of controllers, both of which meet this requirement.

4.3.1 Monitor-Based RHAC. The first controller is directly related to the estimates of the uncertainty monitor. Ideally, one expects to completely cancel the impact of the uncertainty, in which case $x(t)$ would be exactly the same as $x_{\mathrm{id}}(t)$. However, this approach may not be practical since the uncertainty cannot be directly measured. Instead, we can use the low-pass filtered uncertainty $\hat{\sigma}_F(t)$, provided by the monitor in Equation (8), to partially cancel the uncertainty within the bandwidth of the low-pass filter, which is similar to the idea in $\mathcal{L}_1$ adaptive control (Hovakimyan and Cao 2010). Following this idea, the control input of the RHAC can be defined by

$$u(t) = Kx(t) - \hat{\sigma}_F(t), \tag{20}$$

where the feedback gain K is defined in Equation (13). Let T^* be the moment when the RHAC is activated. Assume that at $t = T^*$ the physical dynamics are $\mathcal{P}_j$. Thus, the sequence of dynamics after $t = T^*$, which is $\mathcal{P}_j, \mathcal{P}_{j+1}, \ldots, \mathcal{P}_N$, is controlled by the RHAC. Letting

$$\delta_1 = \|H(s)(BF(s) - \mathbb{I})\|_{\mathcal{L}_1} L_{\max} \tag{21}$$

$$L_{\max} = \max_{k=0,1,\ldots,N} l_k, \tag{22}$$

$$b_{\max} = \max_{k=0,1,\ldots,N} b_k, \tag{23}$$

$$H(s) = (s\mathbb{I} - A_m)^{-1}, \tag{24}$$

$$F(s) = C_z(s\mathbb{I} - A_z)^{-1}B_z, \tag{25}$$

$$\rho_{\mathrm{id}} = \frac{x^\top(T^*)Px(T^*)}{\lambda_{\min}(P)}, \tag{26}$$

the safety of the RHAC-controlled system is established in the following theorem.

Theorem 4.6. *Consider the sequence of dynamics after $t = T^*$, $\mathcal{P}_j, \mathcal{P}_{j+1}, \ldots, \mathcal{P}_N$, controlled by the RHAC defined in Equations (8) and (20). Suppose that Assumptions 3.1 and 3.2 hold. If $x(T^*) \in S$ and the inequality*

$$\rho_{\mathrm{id}} + \frac{\delta_1 b_{\max}}{L_{\max}} < \frac{1 - \delta_1}{\lambda_{\max}(P)} \tag{27}$$

holds, then $x(t) \in \Phi$ and

$$\|x(t) - x_{\mathrm{id}}(t)\| \le \varepsilon = \frac{\delta_1(\rho_{\mathrm{id}} + b_{\max}/L_{\max})}{1 - \delta_1} \tag{28}$$

hold for any $t \in [T^*, \infty)$.

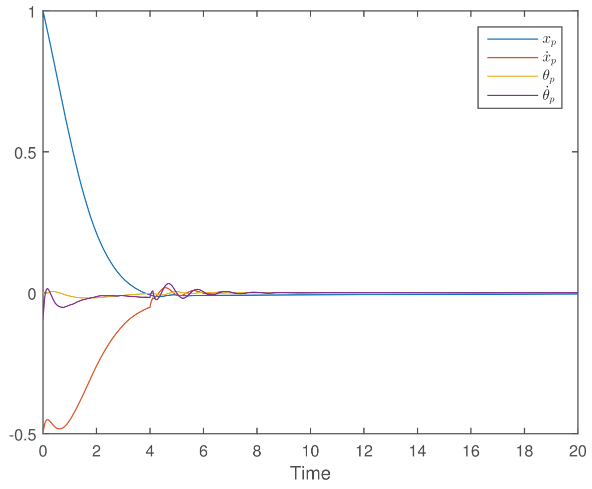

Fig. 5. The state trajectories of the inverted pendulum in the presence of a physical failure.

Remark 4.7. With the definition of ε in Equation (28), the sufficient condition in Equation (27) in fact implies the satisfaction of inequality (Equation (18)) in Lemma 4.5. This condition can serve as a guideline for the selection of K and the low-pass filter $F(s)$.

Remark 4.8. The condition in Equation (27) requires δ_1 to be small. It can always be satisfied by increasing the bandwidth of the low-pass filter in $F(s)$ and/or adjusting the feedback gain K that pushes the eigenvalues of A_m away from the origin (Hovakimyan and Cao 2010). From control design perspective, however, the bandwidth and the feedback gain cannot be too large. High filter bandwidth may degrade the stability margin (Hovakimyan and Cao 2010). High gain K might lead to a matrix P that makes the ellipticity of Φ fairly large and therefore results in a small or even empty S (see inequality Equation (18)). With high bandwidth and high gain, the condition in Equation (27) may lead to a conservative design with limited robustness and a small safety envelope.

Remark 4.9. In the current setting, we directly use the output of the uncertainty monitor to compute the control input. A different way is to include an independent estimator in the RHAC that has the same structure as the monitor but different parameters. The parameters in the RHAC must satisfy the condition, while the monitor does not have to. In that case, we have more flexibility to tune the parameters of the monitor.

Following Example 4.3, we apply the monitor-based RHAC to the system. The state trajectories are plotted in Figure 5. We can see that, after $t = 4$, the states slightly oscillate for a short period because the physical failure is detected and the RHAC is turned on. But the state constraints are still satisfied. After a short adjustment time, the states converge to the origin in the presence of the physical failure.

4.3.2 $\mathcal{L}_1$-Based RHAC. This subsection introduces the $\mathcal{L}_1$-based RHAC. The RHAC is called "$\mathcal{L}_1$-based" because it adopts an $\mathcal{L}_1$ adaptive control architecture (Hovakimyan and Cao 2010), which consists of four components: the state predictor, the adaptation law, the low-pass filter, and the nominal feedback, as shown in Figure 6.

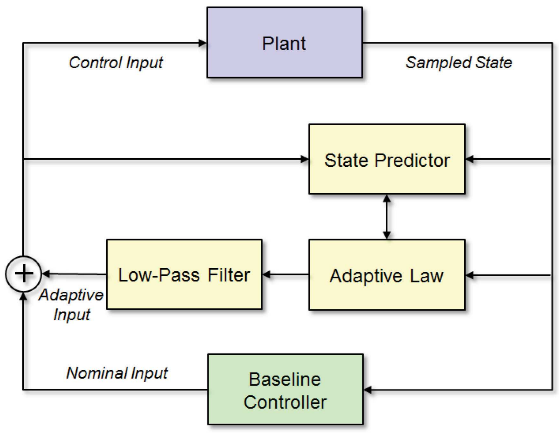

Fig. 6. The $\mathcal{L}_1$ adaptive control structure.

The state predictor in the RHAC is

$$\dot{\hat{x}}(t) = Ax(t) + Bu(t) + \hat{\sigma}(t) - \alpha\tilde{x}(t)$$
$$\hat{x}(T^*) = x(T^*), \tag{29}$$

where $T^* \in \mathbb{R}_0^+$ is the switching moment from the HPC to the RHAC and α is an arbitrary positive constant. The signal $\hat{\sigma}(t)$ is the estimate of $\sigma(t)$ (defined in Equation (9)), generated by the following adaptation law

$$\dot{\hat{\sigma}}(t) = \Gamma\,\mathbf{Proj}_\Psi\left(\hat{\sigma}(t), -\tilde{x}\right), \tag{30}$$

where $\Gamma \in \mathbb{R}_0^+$ is the adaptation gain and

$$\Psi = \left\{\sigma \in \mathbb{R}^n \mid \|\sigma\| \leq \rho_\sigma = \frac{L_{\max}}{\lambda_{\min}(P)} + b_{\max}\right\}. \tag{31}$$

The projection-based operator $\mathbf{Proj}_\Psi : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is defined as follows:

$$\mathbf{Proj}_\Psi(x,y) = \begin{cases} y & \text{if } f(x) \leq 0 \\ y & \text{if } f(x) > 0 \text{ and } \nabla f(x) y \leq 0 \\ y - \frac{f(x)\nabla f(x)^\top \nabla f(x) y}{\|\nabla f(x)\|^2} & \text{if } f(x) > 0 \text{ and } \nabla f(x) y > 0, \end{cases} \tag{32}$$

where $f(x) = \frac{x^\top x - \rho_\sigma^2 + 1 - \mu}{1-\mu}$, and μ is an arbitrarily chosen constant in $(1-\rho_\sigma^2, 1)$. Notice that $f(x) = 1$ implies $x \in \partial\Psi$. We can see that if $x \in \Phi$, where Φ is defined in Equation (15), then the uncertainty $\sigma(t)$ always stays in Ψ by Assumption 3.1. According to Pomet and Praly (1992), the projection operator $\mathbf{Proj}_\Psi$ can ensure $\hat{\sigma}(t) \in \Psi$. More details on the projection-based operator can be found in Pomet and Praly (1992).

Based on the $\mathcal{L}_1$ adaptive control theory, the adaptive control input $u_{\text{ad}}(t)$ is given by

$$\dot{z}_F(t) = A_F z_F(t) + B_F\hat{\sigma}(t)$$
$$u_{\text{ad}}(t) = C_F z_F(t), \quad z_F(T^*) = 0,$$

where (A_F, B_F, C_F) is the state space realization of an $m \times n$ matrix of low-pass filters that are stable and strictly proper with the transfer function

$$D(s) = C_F(s\mathbb{I} - A_F)^{-1}B_F, \tag{33}$$

and z_F is the state of $D(s)$. The control input is given by

$$u(t) = Kx(t) - u_{\mathrm{ad}}(t), \tag{34}$$

where the adaptive input $u_{\mathrm{ad}}(t)$ partially cancels the uncertainty within the bandwidth of the low-pass filter.

Letting

$$\delta_2 = \|H(s)(BD(s) - \mathbb{I})\|_{\mathcal{L}_1} L_{\max},$$

the convergence of the closed-loop system after $t = T^*$ is established in the following theorem.

Theorem 4.10. *Consider the sequence of dynamics after $t = T^*$, $\mathcal{P}_j, \mathcal{P}_{j+1}, \ldots, \mathcal{P}_N$, controlled by the $\mathcal{L}_1$-based RHAC defined in Equations (29)–(34). Suppose that Assumptions 3.1 and 3.2 hold. If $x(T^*) \in S$ and the inequality*

$$\varepsilon + \rho_{\mathrm{id}} < \frac{1}{\lambda_{\max}(P)} \tag{35}$$

holds, where

$$\varepsilon = \frac{\delta_2\left(\rho_{\mathrm{id}} + \frac{b_{\max}}{L_{\max}}\right) + \|H(s)BD(s)(s+\alpha)\|_{\mathcal{L}_1}\sqrt{\frac{\beta}{\Gamma}}}{1-\delta_2} > 0,$$

$$\beta = 4\rho_\sigma^2 + \frac{4\alpha\rho_\sigma^2 + 2\rho_\sigma L_{\max}}{\alpha}\left(\frac{1}{1-e^{-2\alpha T}} + 1\right),$$

and ρ_σ is defined in Equation (31), then $x(t) \in \Phi$ and

$$\|x(t) - x_{\mathrm{id}}(t)\| \leq \varepsilon \tag{36}$$

hold for any $t \in [T^*, \infty)$.

Remark 4.11. The reason for using the $\mathcal{L}_1$ adaptive controller as the RHAC is that it can provide predictable transient performance even in the presence of physical failures. Such transient performance is characterized by the ideal model in Equation (13). With this property, the RHAC can control different faulty dynamics with the same safety envelope discussed in Section 4.2. Meanwhile, the $\mathcal{L}_1$ adaptive controller has the architectural benefit of avoiding model inversion, which enables the system to follow a broader class of ideal models that are non-minimum phase (Che and Cao 2012).

Remark 4.12. When the uncertainties satisfy the matching condition (i.e., $f_k(x,t) = B\hat{f}_k(x,t)$, where $\hat{f}_k : \mathbb{R}^n \times \mathbb{R}_0^+ \to \mathbb{R}^m$ is unknown), Theorem 4.10 will still hold but with a slightly different definition of $\delta_2 = \|H(s)B(D(s) - \mathbb{I})\|_{\mathcal{L}_1} L_{\max}$. With this new definition, the value of δ_2 in the condition given in Equation (35) can be rendered arbitrarily small by increasing the bandwidth of $D(s)$ (without the necessity of tuning K) (Hovakimyan and Cao 2010). It also implies that the error $\|x(t) - x^{\mathrm{id}}(t)\|$ can be arbitrarily small. On the other hand, however, increasing the bandwidth of $D(s)$ may lead to a poor stability margin as mentioned in Remark 4.8, in which case the system may lose robustness (Cao and Hovakimyan 2010).

4.4 Safety Analysis

This Section studies the safety of the RSimplex architecture. The RSimplex running mechanism is shown as follows.

	Running Mechanism I for RSimplex
1	Monitoring the signals $x(t)$ and $\hat{\sigma}_F(t)$;
2	If $x(t)$ satisfies the conditions in (19), go to Step 4; Otherwise, go to Step 3;
3	If $\hat{\sigma}_F(t)$ violates the bound in (12), go to Step 4; Otherwise, go to Step 1;
4	Switch to the monitor-based RHAC (or the $\mathcal{L}_1$-based RHAC).

The safety can be discussed for two cases: switches triggered by Rule I or Rule II. If the switch is triggered by Rule I, we know that, at the switching moment, the state is inside the envelope S. Therefore, the initial states of the system controlled by the RHAC and the ideal model are inside S. By the definition of the ideal model, we know that the ideal state $x^{\text{id}}(t)$ will always stay inside S from the switching moment. Since $x(t)$ follows $x^{\text{id}}(t)$ with a bias ε according to Theorem 4.6 (Theorem 4.10 if using $\mathcal{L}_1$-based RHAC), the actual state $x(t)$ must remain inside Φ and therefore inside Ω (but not necessarily inside S because of the bias ε). When physical failures happen, even if the software has no faults, we need to switch the controller to the RHAC in order to prevent the state from further divergence. In this case, the switch is triggered by Rule II. Note that, at the switching moment, the state is still inside S. We can apply the same analysis as the case of Rule I to conclude safety of the system.

When the RHAC is activated, system stability can also be guaranteed by Theorem 4.6 or Theorem 4.10. Notice that the ideal system in Equation (13) will converge to the equilibrium if $A + BK$ is Hurwitz. Then, by Theorem 4.6 or Theorem 4.10, the actual state will at least converge to a small neighborhood of the equilibrium. The convergence rate can be characterized through the design of feedback gain K in the ideal system since the transient states of the ideal system and the actual system are very close.

4.5 Extended Safety Envelope

The previous sections propose the basic structure of the RSimplex. It guarantees safety in the presence of both physical and software failures. However, we also notice that the safety envelope might be relatively small in some cases. Therefore, we present an approach to extend the volume of the safety envelope. Instead of using a single robust controller as the RHAC, we consider M controller candidates denoted by C_i, $i = 1, 2, \ldots, M$. Each individual C_i is a robust controller. All of these controllers share the same control structure, as shown in Equations (8) and (20), but with different parameters. To be more specific, we use the monitor-based control structure as an example, with the understanding that the analysis also applies to the $\mathcal{L}_1$-based control structure. Among the M controller candidates, an individual controller C_i is defined as follows:

$$\dot{\hat{x}}_i(t) = A_z^i \hat{x}_i(t) - (A_z^i B_z^i + B_z^i A)x(t) - B_z^i B u(t) \tag{37a}$$

$$\hat{\sigma}_i(t) = C_z^i \hat{x}_i(t) + C_z^i B_z^i x(t) \tag{37b}$$

$$\hat{x}_i(T^*) = -B_z^i x(T^*) \tag{37c}$$

$$u(t) = K_i x(t) - \hat{\sigma}_i(t), \tag{37d}$$

where $\hat{x}_i : \mathbb{R}_0^+ \to \mathbb{R}^{l_i}$ is the state of the controller C_i, T^* is the moment when C_i is activated, K_i is the nominal feedback gain for C_i, and (A_z^i, B_z^i, C_z^i) defines the state-space realization of an $m \times n$ matrix of the low-pass filters $F_i(s) = C_z^i(s\mathbb{I} - A_z^i)^{-1}B_z^i$ that are stable and strictly proper.

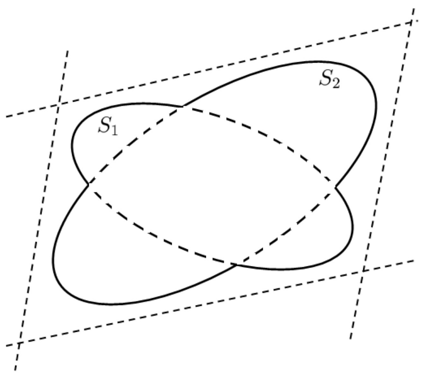

Fig. 7. The combined safety envelope of the L1Simplex architecture.

As indicated in Theorem 4.6, the selection of K_i and $F_i(s)$ must satisfy

$$\|H_i(s)(BF_i(s) - \mathbb{I})\|_{\mathcal{L}_1} L_{\max} < 1 \text{ and}$$
$$A^i_m = A + BK_i \text{ is Hurwitz,}$$

where $L_{\max}$ is defined in Equation (22) and $H_i(s) = (s\mathbb{I} - A^i_m)^{-1}$. Since A^i_m is Hurwitz, there must exist two positive definite matrices $P_i, Q_i \in \mathbb{R}^{n\times n}$ such that

$$P_i A^i_m + \left(A^i_m\right)^\top P_i = -Q_i. \tag{38}$$

Therefore, we are able to define the safety envelope S_i for C_i, following the steps in Section 4.2 (Equation (16)).

Then we can conclude by Theorem 4.6 that if $x(T^*) \in S_i$ and the system is controlled by C_i after the controller switches, we have

$$\|x(t) - x^{\text{id}}(t)\| \le \varepsilon_i$$

with a small $\varepsilon_i > 0$, where $x^{\text{id}}(t)$ satisfies

$$\dot{x}^{\text{id}}(t) = A^i_m x^{\text{id}}(t)$$
$$x^{\text{id}}(T^*) = x(T^*).$$

Notice that with M controller candidates we have M safety envelopes. The overall envelope used in the decision logic is

$$S = \cup_{i=1}^M S_i, \tag{39}$$

as shown in Figure 7. As a result, the decision logic Rule I is to switch the controller to the RHAC when the state is on the boundary of S and about to leave S, where S is defined in Equation (39). The second switch rule remains the same. Once the decision of switching is made, the decision logic checks the region where the current state lies. If $x(T^*) \in S_i$, then the controller switches to C_i. Safety can be guaranteed following the same reasoning in Section 4.4. The running mechanism is stated as follows. Notice that this approach also applies to the traditional Simplex architectures (Sha 2001).

	Running Mechanism II for RSimplex with Extended Safety Envelope
1	Monitoring the signals $x(t)$ and $\hat{\sigma}_F(t)$;
2	If $x(t)$ is on the boundary of S defined in (39) and $x(t)$ is about to leave S, go to Step 4; Otherwise, go to Step 3
3	If $\hat{\sigma}_F(t)$ violates the bound in Equation (12), go to Step 4; Otherwise, go to Step 1;
4	Determine i such that the current state $x(T^*) \in S_i$;
5	Switch to the RHAC, C_i.

5 SIMULATIONS

This section shows the simulation results with the RSimplex architecture. We consider the linearized model of the short-period dynamics of an aircraft (Morelli 1998; Klein and Noderer 1994):

$$\begin{pmatrix} \dot{\alpha}(t) \\ \dot{q}(t) \end{pmatrix} = \underbrace{\begin{pmatrix} Z_\alpha & Z_q \\ M_\alpha & M_q \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} \alpha(t) \\ q(t) \end{pmatrix}}_{x} + \underbrace{\begin{pmatrix} 0 \\ c \end{pmatrix}}_{B} (u(t) + f_0(x,t)),$$

where $\alpha(t)$ is the angle of attack, $q(t)$ is the pitch rate, $u(t)$ is the elevator deflection, and $f_0(x,t)$ describes the exogenous disturbance and model uncertainty. In the simulations, we let

$$f_0(x,t) = 0.2\sin(q(t))\,\alpha(t) + 0.2\cos(5\pi t) + w(t), \tag{40}$$

where $w(t)$ is the disturbance uniformly distributed over $[-0.1, 0.1]$. So $l_0 = 0.5$ and $b_0 = 0.75$. Under the nominal condition, we have

$$A = \begin{pmatrix} -0.1686 & 0.9562 \\ -0.4002 & -0.3393 \end{pmatrix} \text{ and } B = \begin{pmatrix} 0 \\ -1.5313 \end{pmatrix}.$$

It is assumed that the uncertainties due to physical failures satisfy the matching condition. The HPC controller is set to $u = K[\alpha, q]^\top$, where $K = (0.0023, \quad -0.6828)$ and $Q = 0.1\,\mathbb{I}$. Therefore,

$$P = \begin{pmatrix} 0.1416 & 0.0659 \\ 0.0659 & 0.0807 \end{pmatrix}.$$

We first examine the efficiency of the uncertainty monitor and the associated switching Rule I without actually turning on the RHAC. The low-pass filter (A_z, B_z, C_z) is chosen as $A_z = -100\,\mathbb{I}$, $B_z = -A_z$, and $C_z = \mathbb{I}$. The time-varying threshold on $\hat{\sigma}_F(t)$ is computed based on Equation (12). The system runs for 10 seconds and remains stable. Figure 8 plots the history of the norm of the filtered signal of the actual uncertainty $\|\sigma_F(t)\|$ (solid), the norm of the output of the monitor $\|\hat{\sigma}_F(t)\|$ (dash-dot), and the threshold (dash). From the plot we can see that $\|\sigma_F(t)\|$ and $\|\hat{\sigma}_F(t)\|$ are identical, as stated in Theorem 4.1 (in the simulation, the difference between $\sigma_F(t)$ and $\hat{\sigma}_F(t)$ is less than 10^{-13} due to the numerical methods). Another observation is that the threshold is always greater than $\|\hat{\sigma}(t)\|$, which is consistent with the theoretical results in Section 4.2.

We then inject a physical failure at $t = 4$, which changes f_0 to f_1 defined as

$$f_1(x,t) = 0.6\sin(q(t))\,\alpha(t) + 0.6\cos(60\pi t) + w_1(t), \tag{41}$$

and $w_1(t)$ is the disturbance uniformly distributed over $[-0.3, 0.3]$. The system becomes unstable (because the RHAC is off in this simulation). But $\|\sigma_F(t)\|$ and $\|\hat{\sigma}_F(t)\|$ are still identical (their difference in the simulation is still less than 10^{-13}). As shown in Figure 9, the output of the monitor $\|\hat{\sigma}_F(t)\|$ exceeds the threshold right after $t = 4$, which indicates an instant detection of the physical failure using Rule I. We then run this simulation 5000 times with different initial states and different failure injection moments. In 99.9% of the cases, the failure can be detected in 0.2 seconds.

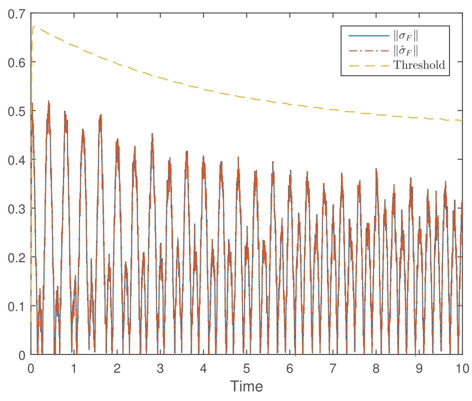

Fig. 8. The time history of the filtered signal of the actual uncertainty $\|\sigma_F(t)\|$ (solid), the output of the monitor $\|\hat{\sigma}_F(t)\|$ (dash-dot), and the threshold (dash), under the uncertainty f_0.

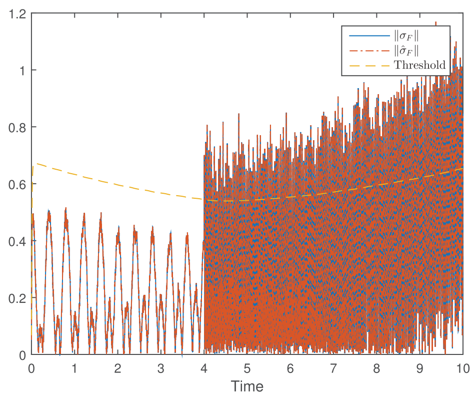

Fig. 9. $\|\sigma_F(t)\|$ (solid), $\|\hat{\sigma}_F(t)\|$ (dash-dot), and the threshold (dash) when injecting a physical failure that changes f_0 to f_1.

In the third simulation, we keep the setting in the previous simulation except $f_0 = 0.2\cos(5\pi t)$ and $f_1 = 0.6\cos(300\pi t)$. The trajectories of $\|\sigma_F(t)\|$, $\|\hat{\sigma}_F(t)\|$, and the threshold are plotted in Figure 10. We see that even when the magnitude of the uncertainty becomes larger, $\|\hat{\sigma}_F(t)\|$ is always smaller than the threshold. This is because the uncertain signal f_1 is filtered by the low-pass filter $F(s)$, given the observation that the bandwidth of the uncertainty is higher than that of $F(s)$. In this case, the monitor-based Rule I fails to detect this specific physical failure.[1] An alternative way to handle this type of physical failures is to set the bandwidth of $F(s)$ high enough. In fact, even if Rule I fails to detect the physical failure and the state diverges, the state will hit the boundary of the safety envelope that triggers switching Rule II. Then the RHAC will be activated and drive the state back to equilibrium. The safety can still be guaranteed by RSimplex, but the settling time is much longer since the failure is not detected instantly.

In the next simulation, instead of physical failures, we inject a software failure at $t = 4$ that makes the HPC generate random control inputs. The system becomes unstable in this case. As shown in Figure 11, $\sigma_F(t)$ and $\hat{\sigma}_F(t)$ in this case are still identical because such identity is independent of the control inputs. However, $\|\hat{\sigma}_F(t)\|$ remains bounded by the threshold despite the software failure. It implies that the monitor-based Rule I cannot detect software failures.

Next, we turn on the switching mechanism and use the $\mathcal{L}_1$-based RHAC in RSimplex. The low-pass filter in the RHAC is $D(s) = \frac{120}{s+120}$ and the adaptation gain is $\Gamma = 10^6$. We inject a software failure at $t_1 = 0.25$ and two physical failures at time $t_2 = 1, t_3 = 2$, respectively. The software failure makes the HPC input randomly generated over $[-150, 150]$. The first physical failure changes f_0 in Equation (40) to f_1 defined in Equation (41). The second physical failure changes f_1 to f_2, where

$$f_2(x,t) = 2\sin(q(t))\,\alpha(t) + 2\cos(200\pi t) + w_2(t),$$

and $w_2(t)$ is uniformly distributed over $[-2, 2]$.

The simulation result is shown in Figure 12. The blue ellipsoid is the ideal safety envelope Φ, the green ellipsoid describes the actual envelope S. The yellow curve is the trajectory of the actual state; the diamond is the initial state, and the crosses represent the occurrence of failures ("SW"

[1]Of course, if the magnitude of f_1 is large enough, $\|\hat{\sigma}_F(t)\|$ will still exceed the threshold since $F(s)$ is not an ideal low-pass filter and cannot completely remove f_1.

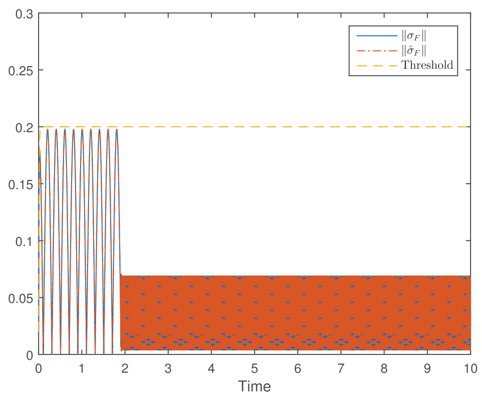

Fig. 10. $\|\sigma_F(t)\|$ (solid), $\|\hat{\sigma}_F(t)\|$ (dash-dot), and the threshold (dash) when injecting a physical failure whose bandwidth is higher than $F(s)$.

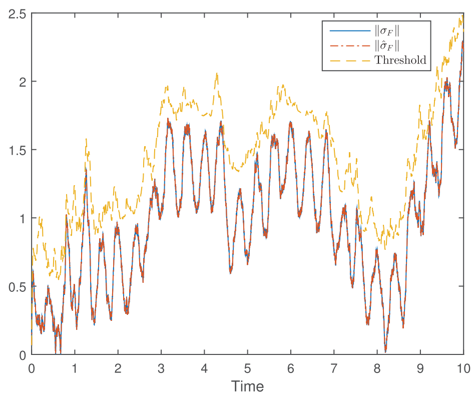

Fig. 11. $\|\sigma_F(t)\|$ (solid), $\|\hat{\sigma}_F(t)\|$ (dash-dot), and the threshold (dash) when injecting a software failure at $t = 4$.

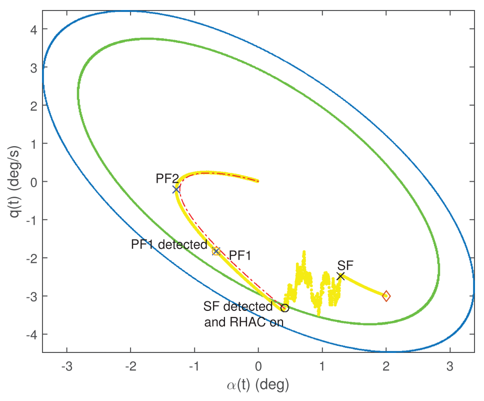

Fig. 12. The state trajectories when experiencing a software failure and two physical failures.

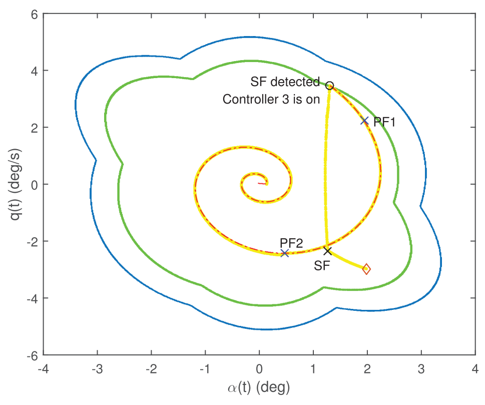

Fig. 13. The state trajectories in RSimplex with an extended safety envelope.

and "PF" stand for "software failure" and "physical failure", respectively). The circles represent the moments when failures are detected. The software failure is detected by Rule II at $t = 0.56$, and the RHAC is turned on at the same moment. The first physical failure is detected right after it happens. Notice that Rule I is not activated under this software failure. This is consistent with our theoretical finding that the monitor is designed for the detection of physical failures. The estimate of the uncertainty only reflects the scale of physical failures, not software failures. We can use this property to distinguish the types of failures in the system.

After the software failure is detected, the RHAC steers the state to equilibrium. The physical failures are compensated by the RHAC and do not affect convergence at all. The red dashed curve is the ideal trajectory generated by the ideal model starting at the moment when the RHAC is turned on. We can see that the actual state closely follows the ideal trajectory under the RHAC, even when physical failures occur. Another observation is that the actual state is temporarily outside the set S after the RHAC is on, but still inside Φ. This is consistent with the theoretical results.

The next simulation studies RSimplex with the extended safety envelope. There are three controller candidates in the RHAC. All of them take the monitor-based control form and share the

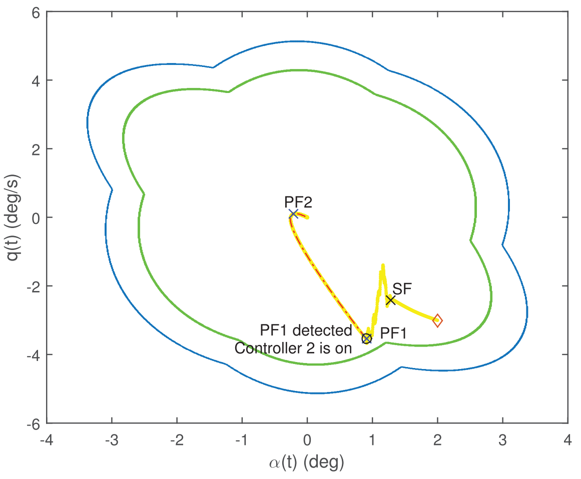

Fig. 14. The state trajectories under software and physical failures, given the first physical failure injected at $t = 0.6$.

same low-pass filter; that is, $F_1(s) = F_2(s) = F_3(s) = \frac{100}{s+100}$. The difference comes from the nominal gain: $K_1 = [0.0023, -0.6928]^\top$, $K_2 = [-0.9036, -1.6927]^\top$, and $K_3 = [-1.1164, -0.0601]^\top$. The ideal safety envelope and the actual envelope are plotted in blue and green, respectively, in Figure 13. Note that the size of the envelope is enlarged. We first consider the same software and physical failures as those in the previous simulation. In this case, after $t = 0.25$ when the software failure happens, the state diverges until it hits the boundary of the ellipsoid related to Controller 3. Then the switch is triggered by Rule II and Controller 3 is activated, which steers the state back to equilibrium despite the physical failures.

Finally, we inject the first physical failure a little earlier (at $t = 0.6$ instead of $t = 1$). We can see from Figure 14 that, before the state hits the boundary of the envelope, the first physical failure is already detected and the switch is triggered by Rule I. Again, it demonstrates the efficiency of the monitor-based switching rule.

6 CONCLUSION

This article presents the RSimplex architecture for safety-critical CPS. It provides an interface to integrate the RFTC techniques into traditional Simplex software architecture, which potentially enables CPS to handle concurrent physical and software failures. The efficiency of RSimplex is shown through rigorous analysis and simulations.

There are still many open problems to be addressed. We assume in this article that sensor and actuator failure will not happen. An interesting extension will be the RSimplex that can handle such failures. Also note that only the static state constraints $a_i^\top x \leq 1$ are considered in this work. When physical failures become serious, these constraints will probably change. How to dynamically adjust the safety envelope in response to such changes will also be studied in future work.

7 PROOFS

7.1 Proof of Theorem 4.1

Proof. For any $t \geq t_0$, there must exist $k \in \{0, 1, \ldots, N\}$ such that $t \in [t_k, t_{k+1})$ with $t_{N+1} = +\infty$. Consider the system in Equation (2) over the time interval $[t_k, t_{k+1})$. The state trajectory $x(t)$

satisfies

$$x(t) = e^{A(t-t_k)}x(t_k) + \int_{t_k}^{t} e^{A(t-\tau)}(Bu(\tau) + f_k(x(\tau),\tau))d\tau. \tag{42}$$

Notice that the state Equation (2) implies $f_k(x(t),t) = \dot{x}(t) - Ax(t) - Bu(t)$. Applying this relation to the system in Equation (10) yields for any $t \in [t_k, t_{k+1})$

$$\begin{aligned} z_\sigma(t) &= e^{A_z(t-t_k)}z_\sigma(t_k) + \int_{t_k}^{t} e^{A_z(t-\tau)}B_z\left(\dot{x}(\tau) - Ax(\tau) - Bu(\tau)\right)d\tau \\ &= e^{A_z(t-t_k)}z_\sigma(t_k) - \int_{t_k}^{t} e^{A_z(t-\tau)}B_z\left(Ax(\tau) + Bu(\tau)\right)d\tau + \int_{t_k}^{t} e^{A_z(t-\tau)}B_z\dot{x}(\tau)d\tau. \end{aligned}$$

Using integration by parts, we have

$$\begin{aligned} z_\sigma(t) &= e^{A_z(t-t_k)}z_\sigma(t_k) - \int_{t_k}^{t} e^{A_z(t-\tau)}B_z\left(Ax(\tau) + Bu(\tau)\right)d\tau \\ &\quad + e^{A_z(t-\tau)}B_z x(\tau)\Big|_{\tau=t_k}^{t} + \int_{t_k}^{t} e^{A_z(t-\tau)}A_zB_zx(\tau)d\tau \\ &= e^{A_z(t-t_k)}z_\sigma(t_k) - \int_{t_k}^{t} e^{A_z(t-\tau)}B_z\left(Ax(\tau) + Bu(\tau)\right)d\tau \\ &\quad + B_zx(t) - e^{A_z(t-t_k)}B_zx(t_k) + \int_{t_k}^{t} e^{A_z(t-\tau)}A_zB_zx(\tau)d\tau \end{aligned} \tag{43}$$

and therefore at t_k, if $k \neq 0$,

$$\begin{aligned} z_\sigma(t_k) &= e^{A_z(t_k-t_{k-1})}z_\sigma(t_{k-1}) - \int_{t_{k-1}}^{t_k} e^{A_z(t_k-\tau)}B_z\left(Ax(\tau) + Bu(\tau)\right)d\tau \\ &\quad + B_zx(t_k) - e^{A_z(t_k-t_{k-1})}B_zx(t_{k-1}) + \int_{t_{k-1}}^{t_k} e^{A_z(t_k-\tau)}A_zB_zx(\tau)d\tau. \end{aligned}$$

Using this equation to replace $z_\sigma(t_k)$ in Equation (43) implies

$$\begin{aligned} z_\sigma(t) = e^{A_z(t-t_k)} &\left(e^{A_z(t_k-t_{k-1})}z_\sigma(t_{k-1}) - \int_{t_{k-1}}^{t_k} e^{A_z(t_k-\tau)}B_z\left(Ax(\tau) + Bu(\tau)\right)d\tau\right) \\ &+ e^{A_z(t-t_k)}\left(B_zx(t_k) - e^{A_z(t_k-t_{k-1})}B_zx(t_{k-1}) + \int_{t_{k-1}}^{t_k} e^{A_z(t_k-\tau)}A_zB_zx(\tau)d\tau\right) \\ &- \int_{t_k}^{t} e^{A_z(t-\tau)}B_z\left(Ax(\tau) + Bu(\tau)\right)d\tau \\ &+ B_zx(t) - e^{A_z(t-t_k)}B_zx(t_k) + \int_{t_k}^{t} e^{A_z(t-\tau)}A_zB_zx(\tau)d\tau. \end{aligned}$$

This equation can be further simplified as:

$$\begin{aligned}
z_\sigma(t) = &e^{A_z(t-t_{k-1})} z_\sigma(t_{k-1}) - \int_{t_{k-1}}^{t_k} e^{A_z(t-\tau)} B_z\left(Ax(\tau) + Bu(\tau)\right) d\tau \\
&+ e^{A_z(t-t_k)} B_z x(t_k) - e^{A_z(t-t_{k-1})} B_z x(t_{k-1}) + \int_{t_{k-1}}^{t_k} e^{A_z(t-\tau)} A_z B_z x(\tau) d\tau \\
&- \int_{t_k}^{t} e^{A_z(t-\tau)} B_z\left(Ax(\tau) + Bu(\tau)\right) d\tau \\
&+ B_z x(t) - e^{A_z(t-t_k)} B_z x(t_k) + \int_{t_k}^{t} e^{A_z(t-\tau)} A_z B_z x(\tau) d\tau \\
= &e^{A_z(t-t_{k-1})} z_\sigma(t_{k-1}) - \int_{t_{k-1}}^{t} e^{A_z(t-\tau)} B_z\left(Ax(\tau) + Bu(\tau)\right) d\tau \\
&- e^{A_z(t-t_{k-1})} B_z x(t_{k-1}) + \int_{t_{k-1}}^{t} e^{A_z(t-\tau)} A_z B_z x(\tau) d\tau + B_z x(t).
\end{aligned}$$

Following a similar recursive procedure, we have

$$\begin{aligned}
z_\sigma(t) &= e^{A_z(t-t_0)} z_\sigma(t_0) - \int_{t_0}^{t} e^{A_z(t-\tau)} B_z\left(Ax(\tau) + Bu(\tau)\right) d\tau \\
&\quad - e^{A_z(t-t_0)} B_z x(t_0) + \int_{t_0}^{t} e^{A_z(t-\tau)} A_z B_z x(\tau) d\tau + B_z x(t) \\
&= - e^{A_z(t-t_0)} B_z x(t_0) + \int_{t_0}^{t} e^{A_z(t-\tau)} \left((A_z B_z - B_z A) x(\tau) - B_z B u(\tau)\right) d\tau + B_z x(t) \\
&= z(t) + B_z x(t)
\end{aligned}$$

for any $t \geq t_0$, given $z_\sigma(t_0) = 0$, where the last equivalence is obtained using Equation (8). Given Equations (8b) and (10b), the proof is completed. □

7.2 Proof of Theorem 4.6

Proof. The proof is based on linear system analysis, which takes two steps: first, we assume $x(t) \in \Phi$ for any $t \geq T^*$ and show the bound on $\|x(t) - x_{\text{id}}(t)\|$; then we show that $x(t)$ is always inside Φ after T^*.

Step 1: Assume that $x(t) \in \Phi$ holds for any $t \geq T^*$. Since Theorem 4.1 shows that $\hat{\sigma}_F(t) = \sigma_F(t)$, we have $u(t) = Kx(t) - \sigma_F(t)$. Applying this input to the system in Equation (2), we have

$$\dot{x}(t) = A_m x(t) - B\sigma_F(t) + \sigma(t),$$

where $\sigma(t)$ is defined in Equation (9).

We now consider $e(t) = x(t) - x_{\text{id}}(t)$. With the preceding equation and Equation (13), the error dynamics are given by

$$\begin{aligned}
\dot{e}(t) &= A_m e(t) - B\sigma_F(t) + \sigma(t) \\
e(t_0) &= 0
\end{aligned}$$

which, with Laplace transform applied to both sides, implies $e(s) = H(s)(\mathbb{I} - BF(s))\sigma(s)$. Therefore,

$$\begin{aligned}\|e(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}} &\leq \|H(s)(\mathbb{I} - BF(s))\|_{\mathcal{L}_1} \|\sigma(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}}\\ &\leq \|H(s)(\mathbb{I} - BF(s))\|_{\mathcal{L}_1} \left(L_{\max}\|x(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}} + b_{\max}\right)\\ &\leq \|H(s)(\mathbb{I} - BF(s))\|_{\mathcal{L}_1} \left(L_{\max}\left(\|x_{\text{id}}(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}} + \|e(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}}\right) + b_{\max}\right),\end{aligned}$$

where the second inequality is based on the assumption $x(t) \in \Phi$ over $[T^*, \infty)$ and Assumption 3.1. Notice that the preceding inequality implies the satisfaction of inequality in Equation (28).

Step 2: We prove $x(t) \in \Phi$ over $[T^*, \infty)$ using a contradiction argument. Suppose that the statement is not true. Since $x(T^*) = x_{\text{id}}(T^*) \in S \subset \Phi$, there must exist at least a time instant $\hat{t}$ after T^* such that

$$x^\top(t)Px(t) < 1, \ \forall t \in [T^*, \hat{t}) \tag{44}$$

$$x^\top(\hat{t})Px(\hat{t}) = 1. \tag{45}$$

Following similar analysis as in Step 1, we know that for any $t \in [T^*, \hat{t})$, the inequality

$$\|x(t) - x_{\text{id}}(t)\| \leq \frac{\delta_1(\rho_{\text{id}} + b_{\max}/L_{\max})}{1 - \delta_1}$$

holds. Since S is an invariant set of the ideal model, $\|x_{\text{id}}(t)\| \leq \rho_{\text{id}}$ for any $t \in [T^*, \hat{t})$. Therefore

$$\begin{aligned}\|x(t)\| &\leq \frac{\delta_1(\rho_{\text{id}} + b_{\max}/L_{\max})}{1 - \delta_1} + \rho_{\text{id}}\\ &= \frac{\rho_{\text{id}}}{1 - \delta_1} + \frac{\delta_1 b_{\max}/L_{\max}}{1 - \delta_1} < \frac{1}{\lambda_{\max}(P)},\end{aligned}$$

where the last inequality comes from the inequality in Equation (27). By continuity of $x(t)$, the inequality also holds for $t = \hat{t}$, which implies

$$x^\top(\hat{t})Px(\hat{t}) \leq \lambda_{\max}(P)\|x(\hat{t})\|^2 < 1.$$

This inequality contradicts the inequality of Equation (45). Thus, we conclude that $x \in \Phi$ for any $t \geq T^*$. □

7.3 Proof of Theorem 4.10

Proof. The proof takes similar steps as that of Theorem 4.6: first, we assume $x(t) \in \Phi$ for any $t \geq T^*$ and show the bound on $\|x(t) - x_{\text{id}}(t)\|$; then we show that $x(t)$ is always inside Φ after T^* with the parameter ε defined in Equation (36).

Step 1: Assume that $x(t) \in \Phi$ holds for any $t \geq T^*$. We first derive the bound on $\|\hat{x}(t) - x(t)\|$ over $t \in [t_k, t_{k+1})$ for any k such that $t_k \geq T^*$. By Equations (2) and (29), the error dynamics of $\tilde{x}(t) = \hat{x}(t) - x(t)$ over $[t_k, t_{k+1})$ satisfy

$$\dot{\tilde{x}}(t) = -\alpha\tilde{x}(t) + \tilde{\sigma}(t) \tag{46}$$

with the initial state $\tilde{x}(t_k)$, where $\tilde{\sigma}(t) = \hat{\sigma}(t) - \sigma(t)$. Consider the Lyapunov function

$$V(\tilde{x}, \tilde{\sigma}) = \tilde{x}^\top\tilde{x} + \Gamma^{-1}\tilde{\sigma}^\top\tilde{\sigma}$$

for these error dynamics. The time derivative of V satisfies

$$\begin{aligned}\dot{V} &= 2\tilde{x}^\top(-\alpha\tilde{x} + \tilde{\sigma}) + 2\Gamma^{-1}\tilde{\sigma}^\top\dot{\tilde{\sigma}}\\ &\leq -2\alpha\tilde{x}^\top\tilde{x} - 2\Gamma^{-1}\tilde{\sigma}^\top\dot{\sigma},\end{aligned}$$

where the second inequality is obtained by the property of the projection-based operator that ensures $\tilde{x}^\top \tilde{\sigma} + \Gamma^{-1}\tilde{\sigma}^\top \dot{\hat{\sigma}} \le 0$. With Assumption 3.1 and the assumption $x(t) \in \Phi$, we know that

$$\|\tilde{\sigma}(t)\| \le 2\rho_\sigma \quad \text{and} \quad \|\dot{\sigma}(t)\| \le L_{\max}.$$

Therefore

$$\begin{aligned}\dot{V} &\le -2\alpha\tilde{x}^\top\tilde{x} + 4\Gamma^{-1}\rho_\sigma L_{\max}\\ &= -2\alpha\left(\underbrace{\tilde{x}^\top\tilde{x} + \Gamma^{-1}\tilde{\sigma}^\top\ \tilde{\sigma}}_{V(t)}\right) + 2\alpha\Gamma^{-1}\tilde{\sigma}^\top\ \tilde{\sigma} + 4\Gamma^{-1}\rho_\sigma L_{\max}\\ &\le -2\alpha V(t) + 4\Gamma^{-1}(2\alpha\rho_\sigma^2 + \rho_\sigma L_{\max}).\end{aligned} \tag{47}$$

Letting $\eta = 4\Gamma^{-1}(2\alpha\rho_\sigma^2 + \rho_\sigma L_{\max})$ and solving this differential inequality for any $t \in [T^*, t_{j+1})$ yields

$$\begin{aligned}V(t) &\le \frac{\eta}{2\alpha}\left(1 - e^{-2\alpha(t-T^*)}\right) + V(T^*)e^{-2\alpha(t-T^*)}\\ &\le \frac{\eta}{2\alpha} + V(T^*),\end{aligned}$$

which means

$$V(t_{j+1}) \le \frac{\eta}{2\alpha} + V(T^*). \tag{48}$$

For any $t \in [t_k, t_{k+1})$ and $k > j$, we know that by solving the inequality in Equation (47) over $[t_k, t_{k+1}]$,

$$V(t) \le \frac{\eta}{2\alpha}\left(1 - e^{-2\alpha(t-t_k)}\right) + V(t_k)e^{-2\alpha(t-t_k)}, \tag{49}$$

which indicates

$$V(t_{k+1}) \le \frac{\eta}{2\alpha} + V(t_k)e^{-2\alpha(t_{k+1}-t_k)} \le \frac{\eta}{2\alpha} + V(t_k)e^{-2\alpha T}, \tag{50}$$

where the second inequality comes from Assumption 3.2. Inequalities in Equations (48) and (50) imply

$$\begin{aligned}V(t_k) &\le \left(\frac{\eta}{2\alpha} + V(T^*)\right)e^{-2\alpha T(k-j-1)} + \frac{\eta}{2\alpha}\frac{1 - e^{-2\alpha T(k-j-1)}}{1 - e^{-2\alpha T}}\\ &\le V(T^*) + \frac{\eta}{2\alpha}\frac{1}{1 - e^{-2\alpha T}}\end{aligned}$$

for $k = j+1, \ldots, N$. Applying this inequality to Equation (49) implies

$$V(t) \le V(T^*) + \frac{\eta}{2\alpha}\left(\frac{1}{1 - e^{-2\alpha T}} + 1\right).$$

Notice that

$$V(T^*) \le \|\tilde{x}(T^*)\|^2 + \frac{4}{\Gamma}\rho_\sigma^2 = \frac{4}{\Gamma}\rho_\sigma^2,$$

since $\tilde{x}(T^*) = 0$. Therefore,

$$V(t) \le \frac{4}{\Gamma}\rho_\sigma^2 + \frac{\eta}{2\alpha}\left(\frac{1}{1 - e^{-2\alpha T}} + 1\right) = \frac{\beta}{\Gamma}$$

for $t \in [T^*, t_N)$, which implies

$$\|\tilde{x}(t)\| \le \sqrt{\frac{\beta}{\Gamma}}. \tag{51}$$

It is easy to verify that for $t \ge t_N$ this bound still holds using similar analysis.

We now go back to the error dynamics in Equation (46). Let $\tilde{\sigma}(s) = \hat{\sigma}(s) - \sigma(s)$. The error dynamics are equivalent to $(s+\alpha)\tilde{x}(s) = \tilde{\sigma}(s)$. It indicates $H(s)BD(s)(s+\alpha)\tilde{x}(s) = H(s)BD(s)\tilde{\sigma}(s)$, where $D(s)$ is the stable low-pass filter in Equation (33). Thus,

$$\begin{aligned}&\|H(s)BD(s)\tilde{\sigma}(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}}\\ &\quad\le \|H(s)BD(s)(s+\alpha)\|_{\mathcal{L}_1}\|\tilde{x}(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}}\\ &\quad\le \|H(s)BD(s)(s+\alpha)\|_{\mathcal{L}_1}\sqrt{\frac{\beta}{\Gamma}},\end{aligned} \tag{52}$$

where the second inequality comes from the inequality in Equation (51).

Consider $x(t) - x^{\text{id}}(t)$. Based on the ideal model (Equation (13)), we have

$$\begin{aligned}x(s) - x^{\text{id}}(s) &= -H(s)BD(s)\hat{\sigma}(s) + H(s)\sigma(s)\\ &= H(s)BD(s)(\sigma(s) - \hat{\sigma}(s)) + H(s)(\mathbb{I} - BD(s))\sigma(s).\end{aligned}$$

Thus,

$$\|x(s) - x^{\text{id}}(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}} \le \underbrace{\|H(s)BD(s)\tilde{\sigma}(s))\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}}}_{\psi_1} + \underbrace{\|H(s)(BD(s) - \mathbb{I})\|_{\mathcal{L}_1}\|\sigma(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}}}_{\psi_2}. \tag{53}$$

With the inequality of Equation (52), we know $\psi_1 \le \|H(s)BD(s)(s+\alpha)\|_{\mathcal{L}_1}\sqrt{\frac{\beta}{\Gamma}}$. Note that $\|\sigma(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}} \le L_{\max}\|x\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}} + b_{\max}$. Then

$$\begin{aligned}\psi_2 &\le \|H(s)(BD(s) - \mathbb{I})\|_{\mathcal{L}_1}\left(L_{\max}\|x(s)\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}} + b_{\max}\right)\\ &\le \delta_2\left(\|x - x^{\text{id}}\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}} + \|x^{\text{id}}\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}} + \frac{b_{\max}}{L_{\max}}\right)\\ &\le \delta_2\left(\|x - x^{\text{id}}\|_{\mathcal{L}_\infty^{[T^*,\,\tau]}} + \rho_{\text{id}} + \frac{b_{\max}}{L_{\max}}\right).\end{aligned}$$

Applying the bounds on ψ_1 and ψ_2 to the inequality of Equation (53) yields the satisfaction of inequality Equation (36).

Step 2: The proof of $x(t) \in \Phi$ over $[T^*, \infty)$ is similar to that of Step 2 in Theorem 4.6 and is therefore omitted. □

ACKNOWLEDGMENTS

The authors gratefully acknowledge the partial financial support of NSF (CNS-1330077 and CNS-1329870).

REFERENCES

Karl J. Åström and Björn Wittenmark. 2013. *Adaptive Control.* Courier Corporation.

Algirdas Avizienis. 1995. The methodology of n-version programming. *Software Fault Tolerance* 3 (1995), 23–46.

Christel Baier, Joost-Pieter Katoen, and Kim Guldstrand Larsen. 2008. *Principles of Model Checking.* MIT Press, Cambridge.

Stanley Bak, Deepti K. Chivukula, Olugbemiga Adekunle, Mu Sun, Marco Caccamo, and Lui Sha. 2009. The system-level simplex architecture for improved real-time embedded system safety. In *Proceedings of the 15th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS).* IEEE, 99–107.

Stanley Bak, Taylor T. Johnson, Marco Caccamo, and Lui Sha. 2014. Real-time reachability for verified simplex design. In *Proceedings of the 35th IEEE International Real-Time Systems Symposium (RTSS).* IEEE, 138–148.

Chengyu Cao and Naira Hovakimyan. 2010. Stability margins of $\mathcal{L}_1$ adaptive control architecture. *IEEE Transactions in Automatic Control* 55, 2 (2010), 480–487.

Jiaxing Che and Chengyu Cao. 2012. $\mathcal{L}_1$ adaptive control of system with unmatched disturbance by using eigenvalue assignment method. In *Proceedings of IEEE Conference on Decision and Control.* IEEE, 4823–4828.

Liming Chen and Algirdas Avizienis. 1978. N-version programming: A fault-tolerance approach to reliability of software operation. In *Proceedings of the 8th IEEE International Symposium on Fault-Tolerant Computing (FTCS).* IEEE, 3–9.

Edmund M. Clarke, Orna Grumberg, and Doron Peled. 1999. *Model Checking.* MIT Press.

Tanya L. Crenshaw, Elsa Gunter, Craig L. Robinson, Lui Sha, and P. R. Kumar. 2007. The simplex reference model: Limiting fault-propagation due to unreliable components in cyber-physical system architectures. In *Proceedings of the 28th IEEE International Real-Time Systems Symposium (RTSS).* IEEE, 400–412.

Lennart Harnefors and Hans-Peter Nee. 1998. Model-based current control of AC machines using the internal model control method. *IEEE Transactions on Industry Applications* 34, 1 (1998), 133–141.

Naira Hovakimyan and Chengyu Cao. 2010. $\mathcal{L}_1$ *Adaptive Control Theory.* SIAM, Philadelphia, PA.

Naira Hovakimyan, Chengyu Cao, Evgeny Kharisov, Enric Xargay, and Irene M. Gregory. 2011. $\mathcal{L}_1$ adaptive control for safety-critical systems. *IEEE Control Systems Magazine* 31, 5 (October 2011), 54–104.

Vladislav Klein and Keith D. Noderer. 1994. *Modeling of Aircraft Unsteady Aerodynamic Characteristics.* Technical Memorandum 109120. NASA LaRC, Hampton, VA.

Xue Liu, Hui Ding, Kihwal Lee, Qixin Wang, and Lui Sha. 2008. ORTEGA: An efficient and flexible software fault tolerance architecture for real-time control systems. In *Proceedings of the Euromicro Conference on Real-Time Systems.* IEEE, 125–134.

Sibin Mohan, Stanley Bak, Emiliano Betti, Heechul Yun, Lui Sha, and Marco Caccamo. 2013. S3A: Secure system simplex architecture for enhanced security and robustness of cyber-physical systems. In *Proceedings of the 2nd ACM International Conference on High Confidence Networked Systems.* ACM, 65–74.

Eugene A. Morelli. 1998. Global nonlinear parametric modeling with application to F-16 aerodynamics. In *Proceedings of American Control Conference*, Vol. 2. IEEE, Philadelphia, PA, 997–1001.

Jean-Baptiste Pomet and Laurent Praly. 1992. Adaptive nonlinear regulation: Estimation from the Lyapunov equation. *IEEE Trans. Automat. Control* 37, 6 (1992), 729–740.

Danbing Seto and Lui Sha. 1999. *An Engineering Method for Safety Region Development.* Technical Report. CMU SEI.

Enrique Ferreira Seto, Danbing and Theodore F. Marz. 2000. *Case Study: Development of a Baseline Controller for Automatic Landing of an F-16 Aircraft Using Linear Matrix Inequalities (LMIs).* Technical Report. No. CMU/SEI-99-TR-020.

Lui Sha. 1998. Dependable system upgrade. In *Proceedings of the Real-Time Systems Symposium.* IEEE, 440–448.

Lui Sha. 2001. Using simplicity to control complexity. *IEEE Software* 18, 4 (2001), 20–28.

Hyungbo Shim and Nam H. Jo. 2009. An almost necessary and sufficient condition for robust stability of closed–loop systems with disturbance observer. *Automatica* 45, 1 (2009), 296–299.

Eduardo D. Sontag and Yuan Wang. 1995. On characterizations of the input-to-state stability property. *Systems & Control Letters* 24, 5 (1995), 351–359.

Xiaofeng Wang, Naira Hovakimyan, and Lui Sha. 2013. L1Simplex: Fault-tolerant control of cyber-physical systems. In *Proceedings of the ACM/IEEE 4th International Conference on Cyber-Physical Systems.* ACM, 41–50.

Jianguo Yao, Xue Liu, Guchuan Zhu, and Lui Sha. 2013. NetSimplex: Controller fault tolerance architecture in networked control systems. *IEEE Transactions on Industrial Informatics* 9, 1 (2013), 346–356.

Received November 2015; revised May 2017; accepted June 2017